Exercise 1: L1 Regularization

For a design matrix $\mathbf{X} \in \mathbb{R}^{n \times p}$ and the vector of targets $\mathbf{y} \in \mathbb{R}^n$, consider Lasso regression, i.e.,

$$\arg\min_{\boldsymbol{\theta} \in \mathbb{R}^p} \underbrace{0.5\|\mathbf{X}\boldsymbol{\theta} - y\|_2^2 + \lambda\|\boldsymbol{\theta}\|_1}_{=:\mathcal{R}_{\text{reg}}} .$$

where $\lambda > 0$ is the regularization parameter.

(a) Since there exists no analytical solution to Lasso regression in general, we want to find a procedure similar to gradient descent that should converge to the true solution.

(i) Explain why $\mathcal{R}_{\text{reg}}$ is not differentiable

(ii) Show that $\mathcal{R}_{\text{reg}}$ is convex
Hint: The sum of convex functions is convex

(iii) Find $\rho_j, z_j \in \mathbb{R}$ for which

$$0.5 \frac{\partial}{\partial \theta_j} \|\mathbf{X}\boldsymbol{\theta} - y\|_2^2 = -\rho_j + \theta_j z_j.$$

(iv) In this situation, we can use the so-called subderivative (for further information see here) which we denote with ∂f for a real-valued convex continuous function f. The subderivative maps a point $\theta \in \mathbb{R}$ to an interval

- and if f is differentiable at $\tilde{\theta} \in \mathbb{R}$, then $\partial f(\tilde{\theta}) = \left\{ \frac{d}{d\theta} f(\tilde{\theta}) \right\}$,
- and for $f(\theta) = \lambda|\theta|$ and $\lambda > 0$, it holds that $\partial f(\theta) = \begin{cases} \{-\lambda\} & \text{for } \theta < 0 \\ [-\lambda, \lambda] & \text{for } \theta = 0 \\ \{\lambda\} & \text{for } \theta > 0 \end{cases}$,
- and for f, g real-valued convex functions with $\partial f(\tilde{\theta}) = [a, b], \partial g(\tilde{\theta}) = [c, d]$,

$$\partial(f + g)(\tilde{\theta}) = [a + c, b + d]$$

where $b \geq a$ and $d \geq c$.

With this compute the subderivative of $\mathcal{R}_{\text{reg},\boldsymbol{\theta}_{\neq j}}$ w.r.t. θ_j, i.e.,

$$\partial_{\theta_j} \mathcal{R}_{\text{reg},\boldsymbol{\theta}_{\neq j}}$$

where $\mathcal{R}_{\text{reg},\boldsymbol{\theta}_{\neq j}} : \mathbb{R} \to \mathbb{R}_{\geq 0}, \theta_j \mapsto \mathcal{R}_{\text{reg}}(\theta_1, \ldots, \theta_j, \ldots, \theta_p)$ for constant $\boldsymbol{\theta}_{\neq j} = (\theta_1, \ldots, \theta_{j-1}, \theta_{j+1}, \ldots, \theta_p)^\top$.
Hint: Use (a)iii

(v) For a real-valued convex function f, the global minimum (if it exists) can be characterized in the following way:

A point $\theta^* \in \mathbb{R}$ is the global minimum of f if and only if $0 \in \partial f(\theta^*)$.

With this show that $\theta_j^* \in \arg\min_{\theta_j \in \mathbb{R}} \mathcal{R}_{\text{reg},\boldsymbol{\theta}_{\neq j}} \iff \theta_j^* = \begin{cases} \frac{\rho_j + \lambda}{z_j} & \text{for } \rho_j < -\lambda \\ 0 & \text{for } -\lambda \leq \rho_j \leq \lambda \\ \frac{\rho_j - \lambda}{z_j} & \text{for } \rho_j > \lambda \end{cases}$.

(vi) Plot θ_j^* as a function of ρ_j for $\rho_j \in [-5, 5], \lambda = 1, z_j = 1$.
(This function is called soft thresholding operator)

(b) Find for non-singular $\boldsymbol{X}^\top \boldsymbol{X}$ the matrix $\boldsymbol{A} \in \mathbb{R}^{p \times p}$ for which

$$\boldsymbol{A}^\top \boldsymbol{X}^\top \boldsymbol{X} \boldsymbol{A} = \boldsymbol{I}.$$

Hint: $\boldsymbol{X}^\top \boldsymbol{X}$ is positive definite

(c) For a design matrix with orthonormal columns, i.e., $\mathbf{X}^\top \mathbf{X} = \boldsymbol{I}$, exists an analytical minimizer of the Lasso regression $\hat{\boldsymbol{\theta}}_{\text{Lasso}} = \left(\hat{\theta}_{\text{Lasso},1}, \ldots, \hat{\theta}_{\text{Lasso},p}\right)^\top$ that is given by

$$\hat{\theta}_{\text{Lasso},i} = \text{sgn}\left(\hat{\theta}_i\right) \max\left\{\left|\hat{\theta}_i\right| - \lambda, 0\right\}, \quad i = 1, \ldots, p,$$

where $\hat{\boldsymbol{\theta}} = \left(\hat{\theta}_1, \ldots, \hat{\theta}_p\right)^\top = (\mathbf{X}^\top \mathbf{X})^{-1} \mathbf{X}^\top \mathbf{y}$ is the minimizer of the unregularized empirical risk (w.r.t. the L2 loss).

Under the assumption that $\boldsymbol{X}^\top \boldsymbol{X}$ is non-singular, your colleague proposes to project $\boldsymbol{X}$ with $\boldsymbol{A}$ from (b), i.e., use $\widetilde{\boldsymbol{X}} = \boldsymbol{X}\boldsymbol{A}$ and then apply the analytical solution given here.

(i) Show that this does not generally solve the original Lasso regression
Hint: You only need to check under which condition $\nabla_{\boldsymbol{\theta}} 0.5\|\mathbf{X}\boldsymbol{\theta} - y\|_2^2 = \nabla_{\boldsymbol{\theta}} 0.5\|\mathbf{X}\boldsymbol{A}\boldsymbol{\theta} - y\|_2^2$. The proof can be finished with a subgradient argument regarding stationarity, which you do not have to do.

(ii) How could you adapt the penalty term such that the solution to the projected problem is equivalent to the original Lasso regression? In this case, can we still solve for parameters independently?

(iii) Does the procedure proposed in (c) perform variable selection?

(d) You are given the following code to compare the quality of the projected Lasso regression vs. the regular Lasso regression:

```
library(matlib)
library(ggplot2)
set.seed(2)

proj_orth_lasso <- function(X, y, lambda){
  # compute X_tilde


  X_tilde =

  # compute analytical solution for X_tilde

  theta_star =
  return(c(theta_star))
}

lasso <- function(X, y, lambda, N){
  p = ncol(X)
  theta = rep(1.0, p)
  for(i in seq(N)){
    j = (i %% p)+1



    rho_j =
    z_j =

    theta[j] =
  }
  return(theta)
}
```

```
rmses = data.frame(rmse = numeric(), type = factor())

p = 10
n = 100

num_opt_steps=400

sigma_noise = 0.1
sigma_signal = 1.0

lambda = 1

rmses = data.frame(rmse = numeric(), projected = factor())

for(i in seq(100)){
  X = matrix(rnorm(n*p, sd=sigma_signal), nrow=n)
  theta_true = rnorm(p)
  idx = rbinom(p, 1, 0.7)
  theta_true[which(idx == 1)] = 0

  y = X %*% theta_true + rnorm(n, sd=sigma_noise)

  rmses = rbind(rmses, data.frame(rmse =
                    n/(n-1)*sd(proj_orth_lasso(X, y, lambda) - theta_true),
                    projected=factor("yes", levels=c("yes", "no"))))
  rmses = rbind(rmses, data.frame(rmse =
                    n/(n-1)*sd(lasso(X, y, lambda, num_opt_steps) - theta_true),
                    projected=factor("no", levels=c("yes", "no"))))
}

ggplot(rmses) +
  geom_boxplot(aes(y = rmse, fill = projected)) +
  ylab(expression(sqrt(Sigma~(hat(theta)[j]-theta["j,true"])^2/p)))
```

Complete the missing code of the algorithms and interpret the result.